APPARATUS AND METHOD FOR MANAGING POWER IN COMPUTER SYSTEM

BACKGROUND OF THE INVENTION

1. Field of the Invention

[1] The present invention relates to a computer system mounted with a plurality of devices, and more particularly, to an apparatus and a method for managing power in a computer system.

2. Background of the Related Art

[2] Fig. 1 illustrates a configuration of an apparatus for managing power in computer system. In the following description, the 'power down mode' refers to all kinds of power modes (i.e. standby, suspend, or power off etc.) except for the power on mode (e.g., normal operation mode). The computer system includes an operation system 10, a device driver 20, a bus driver 30, and devices 40. A power mode for respective devices 40, including a network adaptor, a display adaptor, a mouse, a monitor, a disk drive, a digital camera, a scanner, and a sound card, is collectively determined in accordance with a system power mode that is set up by the operation system 10.

[3] For instance, suppose that the operation system 10 sets up the system power mode to the power down mode. Then, the device driver 20 sets up a power mode for a device it has been operating and controlling to the power down mode. On the other hand, if the operation system 10 sets up the system power mode to an operating mode, the device

driver 20 sets up the power mode for the device it has been operating and controlling to the operating mode and supplies power in a normal manner. Thus, the device driver 20 varies the power mode of the device corresponding to the system power mode set up by the operation system 10 to either the operating mode (e.g., D0) or the power down mode (e.g., D1, D2, D3).

[4] For example, the operation system 10 can generate an I/O Request Packet (IRP) that corresponds to the system power mode it has set up, and output the IRP to the device driver 20. Then, the device driver 20 identifies the IRP and generates a Device Power Mode Packet (DPMP) for enabling the power mode of an object (e.g., corresponding) device 40 to the operating mode or the power down mode and outputs the DPMP to the very top end of a driver stack to notify every driver that the device power mode has been changed.

[5] In response to the DPMP, the device 40 operates its hardware and varies the power mode to the power down mode or the operating mode, and transmits the power mode packet to a bus driver 30. The bus driver 30 changes a bus-power mode to the power down mode when every device under a corresponding bus (e.g. PCI, USB, IEEE1394 and the like) is in the power down mode. Hence, when the operation system 10 sets up the system power mode to the power down mode, the power mode of each device changes to the power down mode. In other words, the entire computer system now enters into the power down mode.

[6] As described above, the related art has various disadvantages. In the related art, the power mode for each device is collectively set up in accordance with the system power mode. In other words, once a current system power mode is in the operating mode, every device is in the operating mode. Accordingly, although some of devices might be in an idle state for a prescribed period of time, the power mode for those devices is all in the operating mode. In so doing, a lot of power was consumed unnecessarily and system loading is done ineffectively.

SUMMARY OF THE INVENTION

[7] An object of the invention is to solve at least the above problems and/or disadvantages and to provide at least the advantages described hereinafter.

[8] Another object of the present invention is to provide an apparatus and method for a computer system that can vary a power mode of a plurality of devices or each device in a computer system to each of a first and second power condition.

[9] Another object of the present invention is to provide an apparatus and a method for managing power in a computer system that can vary power mode of each device in a computer system to a power on mode (normal mode) or a power down mode independently from a system power mode set up by an operation system (OS) of the computer.

[10] The foregoing and other objects and advantages can be realized in a whole or in part by providing an apparatus for managing power in a computer system that includes an

operation system configured to set up a power mode of the computer system, wherein the power mode includes at least one of an operating mode and a power down mode, at least one device configured to perform specific functions and operations, at least one device driver configured to control operations on a corresponding device, wherein the device driver is configured to change the power mode of the corresponding device among the at least one of the operating mode and the power down mode, and a filter driver coupled to the operation system, wherein the filter driver is configured to individually control a selected device to operate in the power down mode when the computer system is in the operating mode.

[11] The foregoing and other objects and advantages can be realized in a whole or in part by providing a method that includes operating a computer system in first and second power modes, operating devices in the computer system in the first and the second power modes, and controlling a selected device in the second power mode when the computer system is in the first power mode.

[12] The foregoing and other objects and advantages can be realized in a whole or in part by providing a method for managing power in a computer system includes detecting at least one device that is in the idle state when a power mode of the computer system is in an operating mode, determining idle time of detected devices in the idle state, and changing a power mode of a corresponding device from the operating mode to a power down mode when the idle state is not reset for a predetermined time.

[13] The foregoing and other objects and advantages can be realized in a whole or in part by providing a method for managing power in a computer system that includes setting up a power state of a corresponding device to a power down state, transferring a power control message for changing the corresponding device to an operating state to a device driver before transferring a message received for the corresponding device to the device driver, transferring the received message to the device driver after the corresponding device is changed to the operating state, and setting the corresponding device to the power down state after the device driver completes handling the message.

[14] The foregoing and other objects and advantages can be realized in a whole or in part by providing a method for managing power in a computer system that includes detecting whether a device is in the idle state when a power mode of the computer system is in operating mode, counting idle time for the detected device in the idle state, and changing the power mode of the device from the operating mode to the power down mode when the idle time that has been counted is greater than a predetermined time.

[15] The foregoing and other objects and advantages can be realized in a whole or in part by providing the filter driver equipped with a packet monitoring function detects whether the devices mounted in the computer system are in the idle state. If a device is in the idle state, the power mode of the corresponding device is varied to the power down mode, independent of the system power mode state that has been set up by the operation system in the computer system. In this manner, no more unnecessary power needs to be supplied to devices in the idle state, and it becomes possible to have a more efficient control

over the load of the computer system. Overall, the present invention can be very advantageously used for developing and expanding the computer system more simply by adding the filter driver into the system, without changing a device driver in the computer system.

[16] Additional advantages, objects, and features of the invention will be set forth in part in the description which follows and in part will become apparent to those having ordinary skill in the art upon examination of the following or may be learned from practice of the invention. The objects and advantages of the invention may be realized and attained as particularly pointed out in the appended claims.

BRIEF DESCRIPTION OF THE DRAWINGS

[17] The invention will be described in detail with reference to the following drawings in which like reference numerals refer to like elements wherein:

[18] Fig. 1 is a block diagram illustrating a related art apparatus for managing power in a computer system;

[19] Fig. 2 is a block diagram illustrating a preferred embodiment of an apparatus for managing power in a computer system according to the present invention;

[20] Fig. 3 is a flow chart shown an exemplary loading process for a filter driver and starting a timer when a computer system is booted up;

[21] Fig. 4 is a schematic diagram illustrating exemplary interactions among a user mode, a Kernel mode, a filter driver and a function driver;

[22] Fig. 5 is a flow chart illustrating a filter driver method for managing an operating mode and power down mode of a device according to a preferred embodiment of the present invention;

[23] Fig. 6 is a diagram illustrating a procedure for generating a FIRP;

[24] Fig. 7 is a diagram that illustrates a procedure for setting up a timer for idle detection and for managing power mode thereof according to another embodiment of the present invention;

[25] Fig. 8 is a flow chart illustrating another preferred embodiment of a method for changing a power mode of an object device in a case that a packet data is received from the user mode (application program) to an operating system and an IRP is transmitted from an IO manager;

[26] Fig. 9 is a flow chart illustrating a procedure for handling transmitted IP request packets at a filter driver; and

[27] Fig. 10 is a block diagram that shows a preferred embodiment of the apparatus for managing power in a computer system according to the present invention.

DETAILED DESCRIPTION OF PREFERRED EMBODIMENTS

[28] Fig. 2 is a block diagram illustrating an apparatus for managing power in a computer system according to a preferred embodiment of the present invention. As depicted in Fig. 2, the computer system can include an operation system (OS) 10, a device

driver 20, a bus driver 30, devices 40, and a filter driver 100. The filter driver 100 can serve to expand e.g., adding or modifying, the functions of the device driver 20.

[29] At least one filter driver 100 is preferably included in a driver stack to expand the functions of the device driver 20. For example, when the filter driver 100 added to a top level of the device driver 20 and the system power mode set up by the operation system 10 is in the operating mode, the filter driver 100 can inform this to the device driver 20. However, the present invention is not intended to be so limited to the location of the filter driver or corresponding functions. The filter driver 100 can further transmit or relay data that has been generated in the devices 40 to the operation system 10 via the device driver 20.

[30] Fig. 3 illustrates a flow chart that shows an exemplary loading process onto a filter driver and starting a timer preferably when a computer system is initialized or booted up. Fig. 4 is a schematic diagram illustrating exemplary interactions among a user mode, a Kernel mode (e.g., handled in the operation system), the filter driver 100 and a function driver 45 according to preferred embodiments.

[31] When the system is booted up, a device driver (e.g., device driver 20) is first loaded onto a memory and then a filter driver (e.g., filter driver 100), and an initializing procedure as shown in Fig. 3 can follow. After a process starts in Fig. 3, control passes to step S31. Timeout values that are preferably set up by a user are read out first (step S31).

[32] Then, other basic works or functions for operating the filter driver preferably come next. That is, a space for temporarily storing an I/O Request Packet (IRP) can be assigned, an event call back routine can be set up, a queue can be initialized and then an

interval variable suspend (e.g., 'Suspend' flag) to be used can be initialized (steps S32, S33, and S34).

[33] Further, a timer can be set up. Then, the timer (e.g., time out timer) can be started (step S35). After step S35, the process is complete.

[34] As shown in Fig. 4, exemplary application programs executed or dispatched in the user mode can generate various sound streams, and sound data of Fig. 4 can be converted to an IP request packet (IRP) through an IO manager and eventually transferred to a sound device driver. A sound device driver (e.g., function driver 45) can be composed of a plurality of dispatch routines, including a CREAT Dispatch (IOC) for handling an IRP that is generated when the sound device driver is used, a power dispatch for handling a power IRP for modifying or inquiring power of the device, or a PnP dispatch for handling an IRP involved in the installation, removal and operation of the device.

[35] Three exemplary methods for a filter driver (e.g., filter driver 100) to manage the operating mode and the power down mode of a corresponding device(s) will now be described. However, the present invention is not intended to be so limited.

[36] Fig. 5 shows a first preferred embodiment of a method for operating a filter driver. The filter driver 100 preferably continuously monitors an operating mode of the devices 40 (step S51). If an object device 40 of the devices 40 does not operate longer than a predetermined time and enters into an idle state, accumulates the object devices 40 idle time (steps S52, S53).

[37] If the accumulated idle time is greater than the predetermined (e.g., base) time (step S53), the filter driver 100 generates a control signal, e.g. a FIRP (False IRP), which can be similar to the IRP generated and output by the operation system 10, to preferably enable the power mode of the object device 40 from the operating mode to transition to a desired power mode (e.g., power down mode) even when the system power mode set up by the operation system 10 is the operating mode. The filter driver 100 can transfer the FIRP to the corresponding device driver 20 (step S55).

[38] Receiving the FIRP, the device driver 20 concludes that the power mode of the object device has been varied (e.g., preferably responds similarly to the IRP that is generated in and outputted from the operation system 10). Thus, the device driver 20 can set the power mode for the object device in the idle state to the power down mode, which can reduce or prevent unnecessary power supply to the device in the idle state (step S56).

[39] If the accumulated value of the idle time is not larger than the predetermined value (step S53) the accumulating of the idle time preferably continues. Thus, control preferably jumps back to step S52. If it is determined that the object device is not in the idle state (step S53), preferably the device is operated in a predetermined power mode (step S54).

[40] Fig. 6 is a diagram illustrating an exemplary procedure for generating a control message being a FIRP. As shown in Fig. 6, the operation system generates an IRP corresponding to the system power mode it has set up, and outputs the IRP to an object device (PoRequestPowerIrp, PDEVICE_OBJECT).

[41] Depending on kinds of the Power IRP, the power state of the device is either changed or inquired to set up a desired power state (e.g., D0, D1, D2, and D3) (MinorFunction, Powerstate). Each numeral indicates different power states. For example, D0 indicates 'Power ON'; D1 indicates 'standby'; D2 indicates 'Suspend'; and D3 indicates 'Power off'.

[42] After setting up the power state, the system can be executed in accordance with the new state (CompletionFunction). A FIRP is output in case the system is on for a predetermined period of time.

[43] In Fig. 6, the term 'DeviceObject' denotes a physical driver of an object device, and the 'MinorFunction' is an example of Power IRPs, changing or inquiring the power of the device. In addition, the 'PowerState' refers to a power state, such as, D0, D1, D3, and D4. The 'CompletionFunction' is a callback routine that dispatches after the power state is changed. Lastly, the 'Context' is a factor to be transmitted to the callback routine.

[44] A second method for the filter driver (e.g., filter driver 100) to manage the operating mode and the power down mode of the device will now be described. The filter driver (e.g., the filter driver 100 of Fig. 2, similar to Fig. 4) can operate an idle detection function of a power manager 70 in the operation system, and if the timer is not reset until the predetermined time lapses, changes the power state of an object device or devices. Further description will be provided below with reference to Figs. 7 through 9.

[45] Fig. 7 is a second preferred embodiment of a method for a filter driver to manage a power mode of a plurality of devices. As shown in Fig. 7, the method can reset the timer for idle detection and for managing the power mode thereof.

[46] The power manager (e.g., filter driver) provides a function for resetting the timer and changing a state of a device after the predetermined (or pre-registered) time so as to perform idle detection on each device and manage power for respective devices (e.g., PoRegisterDeviceForIdleDetection). In the function, there is preferably a timeout value in battery mode, a timeout value in performance mode, and a device power state when the time is out. That is, when the object device is designated, the power manager, after the pre-registered time goes by, generates a power IRP for changing the power state to a registered power state on the object device.

[47] Here, the 'DeviceObject' indicates an object device; the 'ConversationIdleTime' indicates a timeout in battery mode; the 'PerformanceIdleTime' indicates a timeout in AC mode; and the 'State' indicates a device power state after the time is out. On the other hand, when the device is in operation, an internal time should be reset and reinitialized using the function provided by the power manager (e.g., PoSetDeviceBusy).

[48] Fig. 8 is a flow chart illustrating a method for changing the power mode of the object device according to the second preferred embodiment. In Fig. 8, the procedure changes the power mode of the object device in a case that a packet data is received from the user mode to the Kernel mode in Fig. 4 and an IRP is transmitted from an IO manager.

[49] After a process starts in Fig. 8, when the packet is entered into a queue, it is automatically dispatched (step S81). The above routine has a low priority in the operation system so it is dispatched after the operation system completes other work with higher priorities. Thus, there could be a number of IRPs in the queue. If the system is in a state just prior to the suspend mode (i.e. Suspend flag=1), the routine is not dispatched because there is no IRP in the queue.

[50] If the device is in the power down mode according to the step S81, the power of the device goes up preferably to the operating mode (e.g., using a filter driver) (step S82), and then a packet is brought from the queue (step S83), which packet is then passed over the device driver of the next step (step S84). Now that the device is being used, the corresponding timer is reset and the device does not go back to the power down mode (step S85).

[51] If there is any packet left over in the queue, the procedure starting from the step S83 is repeated. However, if there is no further packet, the routine is completed. When the routine is not dispatched again until the timer is out, the power manager (e.g., using the filter driver) sends the power down IRP, setting up the device in the power down mode (step S86).

[52] Generally, as shown in Fig. 9, when an IRP is received from an IO manager (e.g., IO manger 50) in the Kernel mode, a routine begins with dispatch routines receiving all kinds of packets (step S91). The process in Fig. 9 can be applicable to each preferred embodiment of systems, methods and filter drivers according to the present invention.

First, it is determined if the received IRP is a power IRP (step S92), if not, the internal variable, i.e. Suspend flag=1, is compared. In other words, it is compared whether the system is in the state just prior to the suspend mode (step S93).

[53] At the result of the comparison (steps S92 and S93), if it is determined that the received IRP is not the power IRP and the internal variable is Suspend flag=0, packets are put in the queue to promote the filter driver to do the normal operation (step S94). Once the packets are put in the queue, a queue handling routine is executed under the operation system (OS).

[54] If it is determined in the step S92 that the received power IRP is the power IRP, another comparison is made whether the packet is a system power IRP for changing the power state (step S95). If it is determined that the received packet has nothing to do with the system power IRP, the next step (e.g., device driver) proceeds (step S96).

[55] However, if the received power IRP is the system power IRP in the step S95, it is compared whether the IRP is in a new state, or S0 (step S97).

[56] If it is in the new state S0, this means that the system is waking up so the internal variable should be cleared, i.e. Suspend flag=0. This indicates that the filter driver turned back to the normal operating mode (step S98).

[57] If the IRP is not in a state of S0 (step S97), this means that the system is entering into the power down mode. In such case, the internal variable is set to be Suspend flag=1, and no further IRP is allowed to the queue (step S99). Any remaining IRPs in the queue are all removed or compressed, and the routine is finished (step S100).

[58] From steps S98 and S100, the process continues to step S96. From steps S94 and S96, the process ends.

[59] A third preferred embodiment of a filter driver can manage the operating mode and the power down mode of the device without the timer. In a filter driver (e.g., filter driver 100) an initial power state of the device 40 or each device is in the power down mode. Then, in the dispatch routine according to the filter driver, the IRP handling routine as the device is used, the power IRP is generated for transitioning the object device in the operation state before the IRP is transferred to the device driver 20. Once the object device is in the operation state, the received IRP is then forwarded to the device driver 20.

[60] The object device preferably does not go back to the power down mode until the device driver completes the IRP handling. Accordingly, when IRPs are received continuously, the device is powered up when the first IRP is received and powered down at the last IRP. In other words, corresponding to the first associated IRP taken from the queue, the device is powered up and when the queue is empty or clear, it is powered down.

[61] Further, each method described above may or may not use the queue. If the queue is not used, and especially when the device to be used in the application in the user mode wakes up to the operating mode from the power down mode, the device may stop its operation temporarily and restart when the power IRP is being handled.

[62] Fig. 10 illustrates a preferred embodiment of an apparatus for managing power in a computer system according to the present invention. For example in Fig. 10, the device driver 20 is a sound driver 200 and the device 40 is a sound card 400 as described

before with reference to Fig. 2. However, the present invention is not intended to be so limited. In case that a random data is received for the sound card 400 from the operation system 10, the apparatus for managing power (e.g., filter driver 100) checks the power mode of the sound card 400.

[63] If the power mode of the sound card 400 is the operating mode, the received data from the operation system 100, while maintaining the operating mode, passes through the sound driver 200 and is transferred to the sound card 400 according to the filter driver where a series of functions and operations corresponding to it are performed.

[64] However, if the power mode of the sound card 400 is the power down mode, the data from the operation system is temporarily stored, and a control signal, such as the FIRP for enabling the power mode of the sound card from the power down mode to the operating mode, is generated, and the FIRP is later transferred to the sound driver 200. In this manner, the power mode of the sound card is varied from the power down mode to the operating mode before transfer of the received data.

[65] After the power mode of the sound card is changed to the operating mode following the above procedure, the temporarily stored data is sent to the sound card 400 via the sound driver 200, and a series of functions and operations corresponding to it are performed in the sound card.

[66] Meanwhile, the filter driver 100 continuously monitors the operation state of the sound card and if it is in the idle state for more than a designated time, accumulates the idle time. In case the accumulated idle time exceeds the predetermined base time,

independent of the system power mode set up by the operation system 10, the FIRP is generated and transferred to the sound driver 200 in order to enable the power mode of the sound card 400 to the power down mode. Based on the FIRP, the sound driver 200 changes the power mode of the sound card 400 in the idle state to the power down mode, which can reduce or prevent any unnecessary power from being supplied to the sound card in the idle state.

[67] Shortly speaking, without changing the sound driver 200 mounted in the computer system, the power mode of each sound card in the idle state can be varied, independently from the system power mode (e.g., from the operating mode to the power down mode) by adding the above-described filter driver 100. On the other hand, if a data is sent from the operation system 10 to the sound card 400 when the sound card 400 being in the power down mode, the filter driver 100 stores the data temporarily and generates the FIRP, requesting to enable the power mode of the sound card from the power down mode to the operating mode and transfers the FIRP to the sound driver 200. When the power mode of the sound card 400 is changed from the power down mode to the operating mode, the filter driver 100 sends the temporarily stored data to the sound card 400 via the sound driver 200, and a series of functions and operations corresponding to it are performed in the sound card 400.

[68] In summary, without modifying the conventional device driver 20 in the computer system, it is possible to enable the power mode of devices in idle state to the power down mode, independent of the system power mode, by adding preferred methods

and preferred embodiments of filter driver 100 to the system. For example, the filter driver 100, using the packet monitoring function, continuously monitors transceived data packets to diverse devices including a network adaptor, a display adaptor, a mouse, a monitor, a disk drive, a digital camera, a scanner or a sound card, and detects whether a corresponding device is in the idle state, and if so, counts the idle time.

[69] The foregoing embodiments and advantages are merely exemplary and are not to be construed as limiting the present invention. The present teaching can be readily applied to other types of apparatuses. The description of the present invention is intended to be illustrative, and not to limit the scope of the claims. Many alternatives, modifications, and variations will be apparent to those skilled in the art. In the claims, means-plus-function clauses are intended to cover the structures described herein as performing the recited function and not only structural equivalents but also equivalent structures.